**ОТЧЁТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 9**

**УРАВНЕНИЯ ПАРАБОЛИЧЕСКОГО ТИПА**

**(Вариант 10)**

*Выполнил студент 3 курса МОиАИС*

*Соколов Арсений*

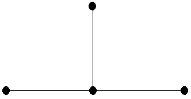
**Задание:**

Решить параболическое уравнение

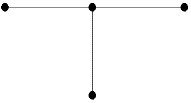


явным методом и неявным методом.

Шаблон для явного метода:



Шаблон для неявного метода:



Вывести результаты в виде двумерных графиков U(x,t).

Неявные схемы решать с помощью прогонки.

Метод прогонки РАСПИСАТЬ подробно!

**Дано:**

[*a*, *b*] = [0; 1],

[*c*, *d*] = [0; 10],

a = 1,

Погрешность решения 0,01 (исходя из погрешности, порядка аппроксимации и условий сходимости для явных схем определить шаги по пространству и времени)

|  |  |  |
| --- | --- | --- |
| № вар. | Начальные условия | Граничные условия |
| 10 |  |  |

**Решение:**

*Схема 1 (явная)*

Разностное уравнение

Начальные условия

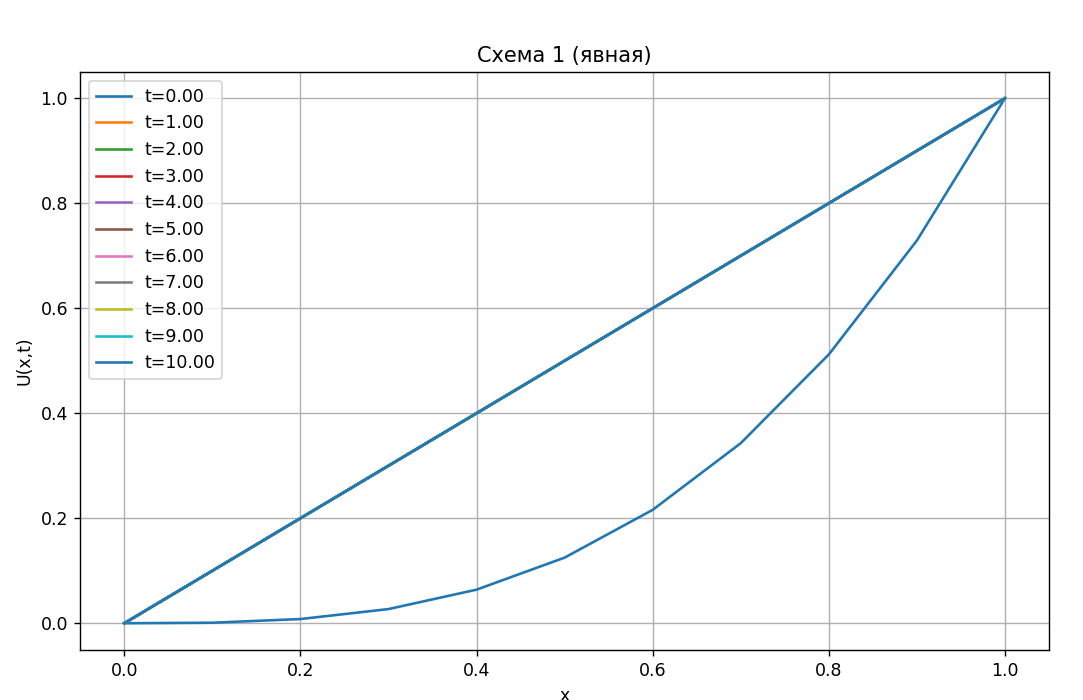
Перепишем уравнение

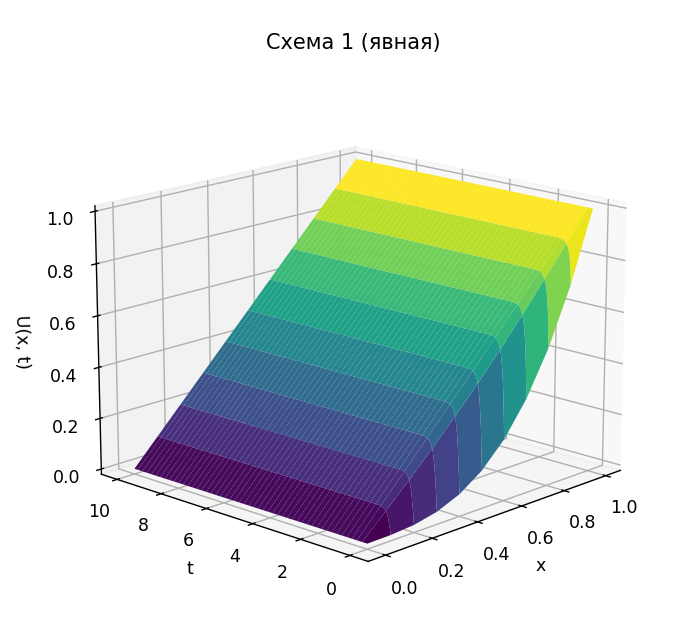
где

Для погрешности 0.01 выберем шаги *τ* и *h* так, чтобы удовлетворять достаточному условию устойчивости

Допустим тогда

Тогда





*Схема 2 (неявная)*

Разностное уравнение

Перепишем уравнение

где

Получим систему линейных уравнений

Систему линейных уравнений будем решать методом прогонки

Определим коэффициенты прогонки и

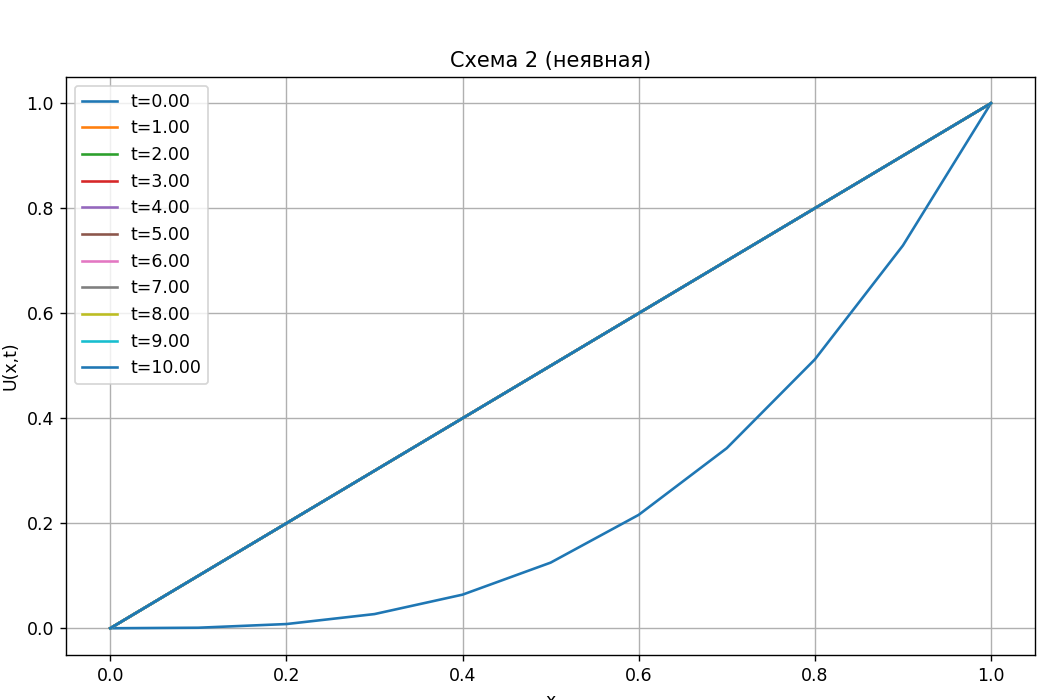
;

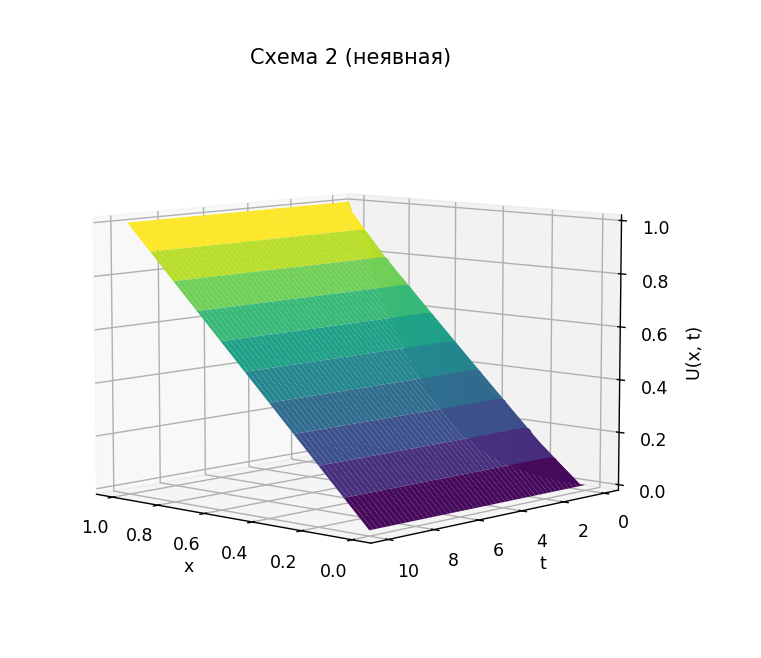
Начальные условия

Граничные условия

;

Тогда





**ПРИЛОЖЕНИЕ 1**

***Программа решения уравнения теплопроводности по схеме 1***

import numpy as np  
import matplotlib.pyplot as plt  
from mpl\_toolkits.mplot3d import Axes3D  
  
# Параметры задачи  
x\_start, x\_end = 0, 1 # Границы по x  
t\_start, t\_end = 0, 10 # Границы по t  
A = 1  
h = 0.1 # шаг по пространству  
tau = 0.005 # шаг по времени  
lambda\_ = A \* tau / h\*\*2  
  
# Проверка на устойчивость  
if lambda\_ >= 1/2:  
 raise ValueError(f"Не выполняется условие устойчивости: lambda = {lambda\_} >= 1/2")  
  
# Число узлов по x и t  
N = int((x\_end - x\_start) / h)  
M = int((t\_end - t\_start) / tau)  
  
# Инициализация сетки  
u = np.zeros((M + 1, N + 1))  
  
# Начальные условия  
x = np.linspace (x\_start, x\_end, N + 1)  
u[0, :] = x\*\*3  
  
# Применение явной схемы  
for j in range(M):  
 for i in range(1, N):  
 u[j+1, i] = u[j, i] + lambda\_ \* (u[j, i+1] - 2\*u[j, i] + u[j, i-1])  
 # Граничные условия  
 u[j+1, 0] = 0  
 u[j+1, N] = 1  
  
# Визуализация результатов в 3D  
T = np.linspace(t\_start, t\_end, M + 1)  
X, T = np.meshgrid(x, T)  
  
fig = plt.figure(figsize=(10, 6))  
ax = fig.add\_subplot(111, projection='3d')  
ax.plot\_surface(X, T, u, cmap='viridis')  
  
ax.set\_xlabel('x')  
ax.set\_ylabel('t')  
ax.set\_zlabel('U(x, t)')  
ax.set\_title('Схема 1 (явная)')  
  
plt.show()  
  
# Визуализация результатов в 2D  
T = np.linspace(t\_start, t\_end, M + 1)  
X, T = np.meshgrid(x, T)  
  
plt.figure(figsize=(10, 6))  
for t in range(0, M+1, M//10):  
 plt.plot(x, u[t, :], label=f't={t\*tau:.2f}')  
plt.xlabel('x')  
plt.ylabel('U(x,t)')  
plt.legend()  
plt.title('Схема 1 (явная)')  
plt.grid(True)  
plt.show()

**ПРИЛОЖЕНИЕ 2**

***Программа решения уравнения теплопроводности по схеме 2***

import numpy as np  
import matplotlib.pyplot as plt  
  
# Параметры задачи  
x\_start, x\_end = 0, 1 # Границы по x  
t\_start, t\_end = 0, 10 # Границы по t  
A = 1 # Коэффициент теплопроводности  
h = 0.1 # Шаг по пространству  
tau = 0.005 # Шаг по времени  
lambda\_ = A \* tau / h\*\*2 # Числовой параметр лямбда  
  
# Проверка на устойчивость  
if lambda\_ >= 1/2:  
 raise ValueError(f"Не выполняется условие устойчивости: lambda = {lambda\_} >= 1/2")  
  
# Число узлов по x и t  
N = int((x\_end - x\_start) / h)  
M = int((t\_end - t\_start) / tau)  
  
# Инициализация сетки  
u = np.zeros((M + 1, N + 1))  
  
# Начальные условия  
x = np.linspace(x\_start, x\_end, N + 1)  
u[0, :] = x\*\*3  
  
# Применение неявной схемы методом прогонки  
for j in range(M):  
 # Вектор правой части  
 b = u[j, 1:-1].copy()  
  
 # Прогоночные коэффициенты  
 alpha = np.zeros(N - 1) # Прогоночный коэффициент alpha  
 beta = np.zeros(N - 1) # Прогоночный коэффициент beta  
  
 # Прогоночные коэффициенты на первом шаге  
 alpha[0] = lambda\_ / (1 + 2 \* lambda\_)  
 beta[0] = b[0] / (1 + 2 \* lambda\_)  
  
 # Прямой ход метода прогонки  
 for i in range(1, N - 1):  
 denom = 1 + 2 \* lambda\_ - lambda\_ \* alpha[i - 1]  
 alpha[i] = lambda\_ / denom  
 beta[i] = (b[i] + lambda\_ \* beta[i - 1]) / denom  
  
 # Обратная подстановка  
 u[j + 1, -2] = beta[-1]  
 for i in range(N - 3, -1, -1):  
 u[j + 1, i + 1] = alpha[i] \* u[j + 1, i + 2] + beta[i]  
  
 # Граничные условия (явные)  
 u[j + 1, 0] = 0  
 u[j + 1, N] = 1  
  
# Визуализация результатов в 3D  
T = np.linspace(t\_start, t\_end, M + 1)  
X, T = np.meshgrid(x, T)  
  
fig = plt.figure(figsize=(10, 6))  
ax = fig.add\_subplot(111, projection='3d')  
ax.plot\_surface(X, T, u, cmap='viridis')  
  
ax.set\_xlabel('x')  
ax.set\_ylabel('t')  
ax.set\_zlabel('U(x, t)')  
ax.set\_title('Схема 2 (неявная)')  
  
plt.show()  
  
# Визуализация результатов в 2D  
plt.figure(figsize=(10, 6))  
for t in range(0, M + 1, M // 10):  
 plt.plot(x, u[t, :], label=f't={t \* tau:.2f}')  
plt.xlabel('x')  
plt.ylabel('U(x,t)')  
plt.legend()  
plt.title('Схема 2 (неявная)')  
plt.grid(True)  
plt.show()